การประยุกต์ใช้ เทอร์โบปาสคาล 4.0 กับ CAI ในวิชาฟิสิกส์

ตอน 3

อาจหาญ สัตยารักษ์

เราได้เรียนรู้คำสั่งด้านกราฟิกของเทอร์โบปาสคาล 4.0 ในตอนที่ 1 และ 2 ไปได้เกือบ 20 คำสั่งแล้ว คราวนี้เราก็จะมาศึกษาคำสั่งต่าง ๆ เพิ่มเติมต่อไป

ก่อนอื่นเรามาหาวิธีที่สะดวกในการสร้างภาพกราฟิกกันก่อน

ปัญหาที่น่าสนใจประการหนึ่งก็คือ รู้จักการใช้คำสั่งด้านกราฟิกเป็นอย่างดี แต่ลำบากใจต่อการกำหนดรูปร่างของภาพตามที่เราต้องการ เช่นจะวาดรูป บ้าน รถยนต์ โทรทัศน์ หรือภาพอื่น ๆ ซึ่งไม่รู้จะกำหนดให้วาดภาพด้วยคำสั่งใดจึงจะได้ภาพตามที่กำหนดไว้

ทางหนึ่งก็คือการเขียนโปรแกรมขึ้นมาช่วยในการวาดรูปเหล่านั้นขึ้นมาแทน ตัวอย่างต่อไปนี้ เป็นการวาดโดยกำหนดรูปร่างเป็นตัวแปรอักขระขึ้นมา ซึ่งเป็นการสะดวกต่อการวาดพอสมควร เพราะเราสามารถมองเห็นรูปร่างอย่างคร่าว ๆ บนจอภาพได้ทันทีที่กำหนดรูปร่างเสร็จ โปรแกรมที่ใช้งานเป็นดังรูปที่ 1 โดยอาศัยอักขระ 0,1,2,3,4 และ * รวม 6 อักขระเป็นตัวกำหนดการวาดภาพ ดังนี้

0 ไม่วาดภาพ

1 วาดรูปสามเหลี่ยมแบบที่ 1 ◥

3 วาดรูปสามเหลี่ยมแบบที่ 3 ◣

* วาดรูปสี่เหลี่ยมจัตุรัส

2 วาดรูปสามเหลี่ยมแบบที่ 2 ◢

4 วาดรูปสามเหลี่ยมแบบที่ 4 ◤

ในโปรแกรมนี้มีคำสั่งใหม่คือ copy ตัวอย่างคือ S:=copy(pic,k,1); ซึ่งตามโปรแกรมเป็นการตรวจสอบอักขระของตัวแปร pic ไปทีละ 1 ตัวตามค่า k ที่เพิ่มขึ้น เพื่อตรวจว่าเป็น 0,1,2,3,4 หรือ * เพื่อจะได้วาดภาพให้ตรงกับความต้องการ ซึ่งเป็นการใช้คำสั่ง Bar,line มาวาดภาพ พร้อมกับสามารถกำหนดขนาดของรูปได้ด้วยตัวแปร Size

รายละเอียดของโปรแกรม Drawpic นั้น หวังว่าคงทำความเข้าใจได้ไม่ยากนัก

ผู้เขียนขอเก็บโปรแกรมในรูปที่ 1 ไปเพิ่มเข้ากับไฟล์ที่เก็บไว้เดิม คือ INITGR.PAS แล้วเก็บไว้ในไฟล์เดิม อันมีผลทำให้ไฟล์ INITGR.PAS นี้มีชุดคำสั่งอยู่ 4 ชุดคำสั่ง คือ OpenGraph, WaitToGo, TextShow, Drawpic ซึ่งจะใช้ทำงานในครั้งต่อไปได้

เมื่อเราได้โปรแกรมต้นแบบสำหรับการวาดภาพไปแล้ว ต่อมาเราก็มาสร้างโปรแกรมเพื่อกำหนดรูปร่างที่สะดวกต่อการวาดภาพของเราต่อไป

วิธีการง่าย ๆ ก็คือกำหนดภาพในตัวแปรมิติ เช่น pix(1) ถึง pix(20) เป็นต้น ซึ่งการกำหนดที่เราจะเห็นบนจอภาพได้ทันที เช่นดังโปรแกรมในรูปที่ 2 ซึ่งกำหนดตัวแปร pix

รูปที่ 1 โปรแกรมสำหรับวาดภาพตามรูปร่างที่กำหนด

```
PROCEDURE Drawpic(x,y,Size :integer;  pic :string);
 var k,Xstart,i:integer;
     s :string[1];
BEGIN
  Xstart :=x;
  for k:=1 to length(pic) do
   begin
     s :=copy(pic,k,1);
     if s ='*' then
         begin
          bar(Xstart,y,Xstart+Size,y-Size); Xstart:=Xstart+Size+1;
         end;
     if s ='0' then Xstart:=Xstart+Size+1;
     if s ='1' then
         begin
           for i :=Xstart to Xstart+Size do
             line(i,y-Size,Xstart+Size,y);
           Xstart:=Xstart+Size+1;
         end;
     if s ='2' then
         begin
           for i :=Xstart to Xstart+Size do
             line(i,y,Xstart+Size,y-Size);
           Xstart:=Xstart+Size+1;
         end;
     if s ='3' then
         begin
           for i :=Xstart to Xstart+Size do
             line(i,y,Xstart,y-Size);
           Xstart:=Xstart+Size+1;
         end;
     if s ='4' then
         begin
           for i :=Xstart to Xstart+Size do
             line(i,y-Size,Xstart,y);
           Xstart:=Xstart+Size+1;
         end;
   end;
END; { Drawpic }
```

รูปที่ 2 โปรแกรมสำหรับการกำหนดรูปร่างของภาพที่ต้องการวาด

```
PROCEDURE Pic(XXstart,YYstart,Size :integer);
var  pix :array[1..20] of string[80];
     no  :integer;
     pic :string[80];
BEGIN
 pix[1] :='0000000000000002*******************300000000000000000000';
 pix[2] :='00000000000000240000000000*00000000130000000000000000000';
 pix[3] :='00000000000002400000000000*00000000013000000000000000000';
 pix[4] :='00000000000024000000000000*00000000001300000000000000000';
 pix[5] :='00000000000240000000000000*00000000000130000000000000000';
 pix[6] :='00000000002400000000000000*00000000000013000000000000000';
 pix[7] :='002*****************************************************00';
 pix[8] :='00*4*****************************************************0*00';
 pix[9] :='00*****************************************************00';
 pix[10]:='******400000001***********************400000001*********';
 pix[11]:='*4****002***3001*********************4002***3001******1*';
 pix[12]:='00000000*****00000000000000000000000000*****00000000000';
 pix[13]:='000000001***40000000000000000000000000001***400000000000';
  for no :=13 downto 1 do
   begin
    pic:=pix[no];
     Drawpic(XXstart,YYstart,Size,pic);
     YYstart:=YYstart-Size-1;
   end;
END; { Pic }
```

เป็นรูปร่างที่เราต้องการ(สามารถแก้ไขจำนวนตัวแปรเพิ่มขึ้นได้)

โดยค่า XXstart, YYstart เป็นตำแหน่งที่ต้องการให้ภาพไปปรากฏ

Size เป็นขนาดของภาพ

เมื่อกำหนดแล้วก็เรียกไปยังโปรแกรม Drawpic เพื่อให้วาดรูปต่อไป ตามรูปนี้จะมองเห็นเป็นรูปรถยนต์ได้อย่างคร่าว ๆ เพื่อการสังเกตที่ดีกว่านี้อาจเปลี่ยนอักขระ '0' ไปเป็น ' ' (Blank) ก็ได้ โดยอย่าลืมเปลี่ยนในโปรแกรม Drawpic ด้วย เพราะที่ผู้เขียนใช้ '0' ก็เพื่อจะได้ดูออกว่ามีตัวอักษรกี่ตัวตอนที่พิมพ์ listing มาให้ศึกษาเท่านั้น

ท่านจะวาดภาพสิ่งของอื่นใดก็ใช้วิธีนี้ได้ทุกครั้ง โดยกำหนดตัวแปรต่าง ๆ (pix,pic และค่า no เริ่มต้น) ให้เหมาะสมก็จะได้ภาพตามที่ท่านต้องการได้ ดังนั้นชุดคำสั่ง pic นี้จึงสามารถดัดแปลงให้ตรงกับภาพที่เราต้องการได้เสมอเมื่อจะทำ CAI ในเรื่องต่อ ๆ ไป

โปรแกรมทั้งหมดที่กล่าวมานี้ยังไม่ปรากฏรูปบนจอภาพ เราต้องมาเขียนส่วนของโปรแกรมหลักที่ใช้งานร่วมกับไฟล์ INITGR.PAS ในแผ่นบันทึก ดังรูปที่ 3 ซึ่งจะได้ภาพดังรูปที่ 4

โปรแกรมจะทำงานได้สมบูรณ์ต้องเรียกไฟล์ INITGR.PAS เข้ามาโดยอาจใช้วิธีเรียกด้วย Ctrl-KR หรือ {$I INITGR.PAS} ก็ได้ โดยต้องมีโปรแกรมในรูปที่ 2 อยู่ด้วยเช่นกัน

คราวนี้เรามาศึกษาคำสั่งเพิ่มเติมอีกดังนี้

MoveRel(Dx,Dy :integer); คำสั่งเพื่อเลื่อนจุดไปยังจุดใหม่โดยเป็นการเลื่อนแบบสัมพัทธ์กับจุดที่เคยลงไว้เดิม (ต่างกับ MoveTo) กล่าวคือถ้าเคยลงจุดที่ 50,60 แล้วมาใช้คำสั่ง MoveRel(30,30); ก็จะตรงไปยังจุด 80,90 โดยไม่มีการลากเส้น

LineRel(Dx,Dy :integer); คล้ายคำสั่ง MoveRel แต่เป็นการลากเส้นตรง เช่น ลงจุดที่ 50,60 แล้วมาใช้ lineRel (30,30) ก็จะเป็นการลากเส้นจาก 50,60 ไปยัง 80,90 จึงต่างกับ LineTO ที่เป็นการลากตรงไปยังจุดที่ระบุ

PutPixel(x,y :integer; Pixel :word); คำสั่งลงจุด ณ จุดที่กำหนดด้วยสีที่ระบุในคำสั่ง เช่น PutPixel(100,50,1); PutPixel(40,30,0);

Bar3D(x1,y1,x2,y2 :integer; Depth :word; Top : Boolean); คำสั่งเขียนกราฟแท่ง 3 มิติ เช่น Bar3D(20, 30,50,120,10,TopOn); เป็นการวาดรูปกราฟแท่งมีเงาที่ด้านข้างและด้านบนโดยเงาหนา 10 จุด หรือ Bar3D(300,70, 350,230,5,TopOff); ไม่มีเงาที่ด้านบน

SetLineStyle(LineStyle, Pattern, Thickness :word); คำสั่งเพื่อสร้างรูปแบบของเส้นที่ลาก โดย

LineStyle มีค่า 0,1,2,3 ได้เส้น 4 แบบ ถ้ากำหนดเป็น 4 จะใช้ตัวแปร Pattern ได้ (0-3 ตัวแปร Pattern จะไม่มีผล)

Pattern ตัวแปรกำหนดรูปแบบ ค่า $00 ถึง $FF (หรือใช้เลขฐานสิบ 0 ถึง 255 ก็ได้ ซึ่งโปรแกรมตัวอย่างจะใช้เลขฐานสิบ) ทั้งนี้ค่าของ LineStyle ต้องเป็น 4 ด้วย

Thickness กำหนดได้ 2 ค่าคือ 1 เป็นเส้นบาง และ 3 จะเป็นเส้นหนา

เช่น SetLineStyle(0,$AA,1); SetLineStyle(4,$55,3);

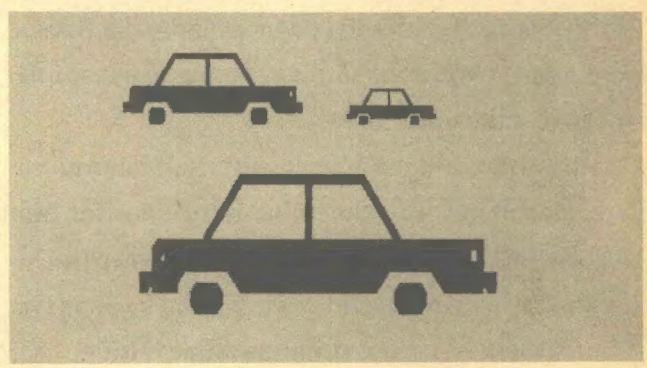
SetFillPattern(Pattern :FillPatternType; color :word); คล้ายกับ SetFillStyle แต่เป็นการกำหนด Pattern ด้วยตัวเราเองโดยมีรูปแบบดังโปรแกรมในรูปที่ 5

FillPoly(NumPoints :word; var PolyPoints); การวาดรูปหลายเหลี่ยมทึบ

รูปที่ 3 โปรแกรมหลักสำหรับนำรูปร่างที่กำหนดมาแสดงบนจอภาพ

```
{ MAIN }
begin
  opengraph;
     pic(50,50,1);
     pic(60,120,3);
     pic(190,50,0);
  waittogo;
  closegraph;
end.
```

รูปที่ 4 ภาพที่ได้จากโปรแกรมในรูปที่ 1 ถึงรูปที่ 3



รูปที่ ๕ โปรแกรมตัวอย่างการใช้คำสั่งกราฟิกต่าง ๆ

```
{$I INITGR.PAS }
const
 Polygon0: array[1..7] of PointType=((x:150; y:50),(x:70; y:75),
        (x:62; y:100),(x:48; y:100),(x:25; y:75),(x:20; y:10),
        (x:150; y:50));
 Polygon1:array[1..4] of PointType=((x:250; y:200),(x:270; y:175),
        (x:200; y:150),(x:250; y:200));
 MyPattern0: FillpatternType =($BB,$BB,$BB,$BB,$BB,$BB,$BB,$BB);
 MyPattern1: FillpatternType =($01,$24,$35,$46,$57,$68,$79,$8A);
 MyPattern2: FillpatternType =($77,$AA,$77,$AA,$77,$AA,$77,$AA);
 MyPattern3: FillpatternType =($FF,$00,$FF,$00,$FF,$00,$FF,$00);
 MyPattern4: FillpatternType =($48,$48,$48,$48,$48,$48,$48,$48);
 var k,i,y :integer;
begin
 opengraph;      y:=110;
 PutPixel(400,10,1);  Moveto(450,20);  MoveRel(50,-18);
  for k:=40 downto 0 do
   begin
    LineRel(-k,0);  delay(25);
   end;
  Bar3D(370,100,390,50,10,TopOn);  Bar3D(420,100,440,20,10,TopOff);
  SetFillPattern(MyPattern0,blue);
  FillPoly(SizeOf(Polygon0) div Sizeof(PointType),Polygon0);
  DrawPoly(SizeOf(Polygon1) div Sizeof(PointType),Polygon1);
  PieSlice(330,150,0,60,50);
  SetFillPattern(MyPattern1,blue);     PieSlice(330,150,60,135,50);
  SetFillPattern(MyPattern2,blue);     PieSlice(330,150,135,223,50);
  SetFillPattern(MyPattern3,blue);     PieSlice(330,150,223,276,50);
  SetFillPattern(MyPattern4,blue);     PieSlice(330,150,276,360,50);
  Rectangle(360,50,290,20);
  for k:=0 to 3 do
   begin
    SetLineStyle(k,$AA,1);
      for i:=y to y+10 do   Line(10,i,120,i);
    Line(10,i+4,120,i+4);
    SetLineStyle(k,$AA,3);  Line(10,i+8,120,i+8);
     y:=y+25;
   end;
  for k:=130 to 255 do
   begin
    SetLineStyle(4,k,1);
      Line(170,k-110,270,k-110);
   end;
 waittogo;
 closegraph;
end.
```

DrawPoly(NumPoints :word; var PolyPoints); การวาดรูปหลายเหลี่ยมโปร่ง

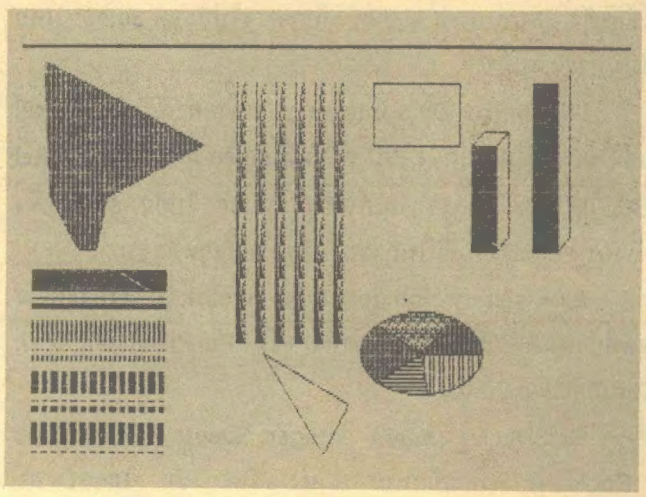
Rectangle(x1,y1,x2,y2 :integer); คำสั่งวาดรูปสี่เหลี่ยมโปร่ง คล้าย ๆ กับ Bar

PieSlice(x,y :integer; StAngle,EndAngle,Radius :word); คำสั่งวาดรูปส่วนย่อย ๆ ของวงกลม หรือวงกลมทั้งวงก็ได้แต่จะเป็นวงกลมทึบ

โดยโปรแกรมในรูปที่ ๕ จะแสดงการใช้คำสั่งเหล่านี้เพื่อเป็นตัวอย่างให้ศึกษา ซึ่งโปรแกรมนี้จะทำงานได้ต้องเรียกไฟล์ INITGR.PAS มาใช้งานด้วย โดยรูปที่ ๖ จะแสดงภาพที่ได้

เราจะมาทำ CAI วิชาฟิสิกส์กันสักหนึ่งเรื่อง โดยคราวนี้ได้เลือกเอาเรื่อง การเคลื่อนที่แบบซิมเปิลฮาร์โมนิก ซึ่ง

รูปที่ ๖ ภาพที่ได้จากการใช้คำสั่งกราฟิกในโปรแกรมรูปที่ ๕



เป็นบทเรียนในชั้นมัธยมปีที่ 5 ภาคเรียนที่ 1 มาทำ โดยอาศัยแนวคิดว่าเมื่อนำรถทดลองมาติดกับสปริงที่ตรึงปลายด้านหนึ่งกับจุดหยุดนิ่ง เมื่อออกแรงให้รถยืดออกแล้วปล่อยรถทดลองนั้นก็จะเคลื่อนที่แบบซิมเปิลฮาร์โมนิก แต่เนื่องจากการทดลองจริง ๆ อาจจะมองเห็นการเคลื่อนที่ไม่ดีนัก เพราะความเร็วเป็นเหตุ

ดังนั้นเครื่องคอมพิวเตอร์จึงต้องมาช่วยเราอีกแรง

โปรแกรมเป็นดังรูปที่ 7 ซึ่งแบ่งเป็น 3 ส่วน คือ ชุดคำสั่ง Spring, SimpleHarmonic และโปรแกรมหลักเพื่อให้เกิดภาพที่กำหนดไว้ ขออธิบายอย่างคร่าว ๆ ดังนี้

ชุดคำสั่ง Spring เป็นการวาดรูปสปริงที่จะยืดหดตามขนาดแรงที่ถูกรถทดลองกระทำ ซึ่งใช้คำสั่งวาดด้วย Ellipse เพื่อให้เกิดรูปวงรีเมื่อค่า x,y เปลี่ยนไป

ชุดคำสั่ง SimpleHarmonic เป็นส่วนสำคัญของโปรแกรม โดยมีการทำงานที่สำคัญดังนี้

MyPattern1-2 กำหนดเพื่อให้การระบายสีรูปแปลกตา ใช้คู่กับ SetFillPattern

Pic เป็นตัวแปรอะเรย์สำหรับเก็บรูปรถทดลองติดสปริง จำนวน 36 รูป ขอให้สังเกตการใช้คำสั่งเกี่ยวกับเรื่องนี้ทั้งหมดให้มาก ๆ เพราะจะนำไปใช้แสดงการเคลื่อนที่ที่นุ่มนวลในการเขียน CAI ของท่านได้เป็นอย่างดี

Size เป็นตัวแปรสำหรับกำหนดขนาดของภาพที่เราจะเก็บให้คู่กับคำสั่ง ImageSize และ GetMem โดยจะเกี่ยวพันกับตัวแปร Pic ด้วย

รูปที่ 7 โปรแกรมวิชาฟิสิกส์แสดงการเคลื่อนที่แบบซิมเปิลฮาร์โมนิก

```
{$I INITGR.PAS}

PROCEDURE Spring(i,time,x,y :integer);
 BEGIN
   repeat
   begin
    Ellipse(i,40,300,250,x,y);  i:=i+x+1;
   end;
   until i>400;
 END; { Spring }

PROCEDURE SimpleHarmonic;
const
 MyPattern1: FillpatternType =($01,$24,$35,$46,$57,$68,$79,$8A);
 MyPattern2: FillpatternType =($77,$AA,$77,$AA,$77,$AA,$77,$AA);
var
 pic    : array[1..36] of pointer;
  p     : pointer;
 Size   :word;
   x ,k :integer;
BEGIN
  SetFillPattern(MyPattern1,blue);
  Bar3D(0,60,640,70,5,TopOff);  SetFillPattern(MyPattern2,blue);
  SetLineStyle(1,$AA,3);  Rectangle(0,0,20,60); Bar(2,2,18,58);
  SetLineStyle(0,$AA,1);  SetFillStyle(6,1);    Bar(400,30,550,50);
  SetFillPattern(MyPattern2,blue);
  PieSlice(420,48,0,360,15);          PieSlice(520,48,0,360,15);
   Size :=ImageSize(400,27,639,60); getmem(p,Size);
   GetImage(400,27,639,60,p^);      SetFillStyle(0,0);
   for x:=1 to 36 do
    begin
     Spring(20,35+x*2,38-x,7);       PutImage(405-x*5,27,p^,0);
     setcolor(0);
      for k:=0 to 3 do
       begin
         PieSlice(425-x*5,48,x*10+k*90,x*10+k*90+3,8);
         PieSlice(525-x*5,48,x*10+k*90,x*10+k*90+3,8);
       end;
     Size :=ImageSize(30,27,550,59);  getmem(pic[x],Size);
     GetImage(20,27,550,59,pic[x]^);
     Bar(20,20,550,59);               SetColor(1);
    end;
  repeat
   begin
```